

# A web-based tool to Analyze Semantic Similarity Networks.

Mario Cannataro, Pietro Hiram Guzzi, Marianna Milano, Pierangelo Veltri

December 24, 2014

## Abstract

In computational biology, biological entities such as genes or proteins are usually annotated with terms extracted from Gene Ontology (GO). The functional similarity among terms of an ontology is evaluated by using Semantic Similarity Measures (SSM). More recently, the extensive application of SSMs yielded to the Semantic Similarity Networks (SSNs). SSNs are edge-weighted graphs where the nodes are concepts (e.g. proteins) and each edge has an associated weight that represents the semantic similarity among related pairs of nodes. The analysis of SSNs may reveal biologically meaningful knowledge. For these aims, the need for the introduction of tool able to manage and analyze SSN arises. Consequently we developed SSN-Analyzer a web based tool able to build and preprocess SSN. As proof of concept we demonstrate that community detection algorithms applied to filtered (thresholded) networks, have better performances in terms of biological relevance of the results, with respect to the use of raw unfiltered networks.

## 1 Introduction

Biological informations about genes and proteins are stored into biological ontologies [3] [8] such as Gene Ontology (GO). GO has gained a wide diffusion in bioinformatics and computational biology since it provides a structured and uniform vocabulary of terms, GO terms, useful to describe a domain of interest [16]. Gene Ontology (GO) [5] organizes a set of GO terms into three main taxonomies: Molecular function (MF), Biological Process (BP), and Cellular Component (CC). Then, each GO Term is related to any number of gene products through the “ annotations process ”. These annotations are available in public databases as, the Gene Ontology Annotation (GOA) database [5]. Annotations allow the comparison of entities focusing on semantics aspects through semantic similarity measures (SSMs) [16] [9]. A SSM takes in input two or more terms of GO and produces as output a numeric value in the [0..1] interval representing their similarity. The use of SSMs to evaluate the functional similarity among gene products is becoming a common task and consequently the use of SSMs

to analyze biological data is gaining a broad interest from researchers [6] [9] . Many analysis methods based on use of SSMs are present in literature, here, we focus on semantic similarity networks [16]. A semantic similarity network of proteins (SSN) is an edge-weighted graph, where the nodes represents the analyzed set of proteins, and the set of edges, with a associated weight, represents the semantic similarity among related pairs of nodes. These networks are constructed by computing some similarity value between genes or proteins and then linking nodes whose similarity is greater than zero. A possible bias consists of the presence of meaningless edge SSN related to high similarity value. Thus, a thresholding preprocessing can be improve the building of SSN. Many methods for networks thresholding exist, for example, methods based on global threshold, or based on local thresholds. However, internal characteristics of SSMs [15] bring to exclude the use of global thresholds, since small regions of relatively low similarities may be due to the characteristics of measures while proteins or genes have high similarity. Whereas the use of local threshold may be influenced by the presence of local noise and in general may cause the presence of biases in different regions. In a previous work we presented novel hybrid thresholding method employing both local and global approaches and based on spectral graph theory. The choice of the threshold is made by considering the highlighting of nearly-disconnected components. The evidence of the presence of these components is analyzed by calculating the eigenvalues of the Laplacian matrix [7] [14]. The choice of this simplification has a biological counterpart on the structure of biological networks. It has been proved in many works that these biological networks tend to have a modular structure in which hub proteins (i.e. relevant proteins) have many connections [1] [12] [20]. Hub proteins usually connect small modules (or communities), i.e. small dense regions with few link to other regions [19] in which proteins share a common function. We here presented a Web Based Tool for Analyzing Semantic Similarity Networks able to build and preprocess SSN. Furthermore, SSN-Analyzer enables the calculation of semantic similarity matrices from a proteins/genes input dataset on which the SSNs are constructed and analyzed. In this way the user can easily perform the testing of interest. The web tool is realized with R Shiny package.

## 2 Semantic Similarity Measures

Semantic Similarity measures are instrument generally applied to Gene ontology Terms (GO terms) in order to quantify the similarity of two or more terms of belonging ontology. Since genes and proteins are annotated by set of GO terms, these measures are carried on genes and proteins. In specific, the SSMs are mathematical functions able to associate a numerical value for each couple of terms of the same ontology quantifying their similarity according to following definition:

$$SS_M : TxT \rightarrow RT : t_1, \dots, t_2, t_i \in O \quad (1)$$

In literature there are more than 40 different semantic similarity measures and many ones rely on concepts from information theory such as information

content (IC). The IC measures how specific and informative is any term according to the frequency of the occurrence of this term into the corpus of annotations considering the Gene Ontology Database:

$$IC(c) = -\log(p(c)) \quad (2)$$

where  $p(c)$  is the probability of the occurrence of  $c$  [11]. Thus, according to above formulation, rare term contains a greater amount of information. The IC-based measures usually take as input two terms and then calculate the IC of a common ancestor of them. The ancestor can be the common ancestor with the maximum value of IC (Maximum Informative Common Ancestor MICA) or disjoint common ancestor (technical DCA). This approach considers all disjoint common ancestors that does not incorporate any other ancestor including the semantic tree structure of the terms. This kind of IC evaluation is also indicated as GraSM (Graph-based Similarity Measure) option.

For example Resnik measure applied [17] to  $t1$  and  $t2$  terms considers the IC of MICA:

$$sim_{Res}(c1, c2) = IC(c_{MICA}) \quad (3)$$

Others examples of measures based on IC of MICA are Lin measure [18] and Relevance measure [13], whereas the GraSM approach can be applied to Resnik, Lin and Jiang-Conrath measures. Also, there are semantic similarity measures based on topological properties of GO DAG.

The Czekanowski-Dice measure [10], for example, calculates the distance of two terms  $t1$  and  $t2$  into the GO structure, and the subsequent mapping of the distance into similarity indices, i.e. the higher distance the lower similarity. The Kappa measure [2] represents each gene in GO as  $n$ -dimensional vector, where each  $n$ -dimension identifies an ontology annotation. The similarity measure among two genes is calculated by taking into account the occurrence of similar annotations in the representative vectors. Similarly, the Cosine similarity [4] represents each gene  $g$  as a  $n$ -dimensional vector in which each component  $i$  represents the information content of the annotation  $i$ . The similarity of two genes is defined as the cosine of the angle between their vectorial representation. Finally the Weighted-Jaccard (also known as SimGIC measure) considers the information content of a group of GO terms. It directly evaluates the similarity between two sets of GO terms  $t1$  and  $t2$  considering the contributions of all the shared ancestors of the two sets. Thus, it can be directly applied to proteins and genes.

### 3 Semantic Similarity Networks

We developed a novel hybrid thresholding method exploiting both local approach, that computes a different threshold for each node or group of nodes and global one, that prune all edges with weights lower than the threshold. Moreover this hybrid thresholding method refers to spectral graph theory. The algorithm examines each node of a input graph and stores the weights of adjacent edge.

Then a threshold  $k = \mu + \alpha \times sd$  is determined, where  $\alpha$  is a global variable, and  $\mu$  and  $sd$  are the average and standard deviation of all weights, i.e. local components.

Then, the algorithm compares each edge weight with threshold  $k$ : if the weight is greater than  $k$  considering the adjacent of both its nodes, it will be inserted into a novel graph with weight 1, whereas if the weight is greater than  $k$  considering only one of its adjacent nodes, it will be inserted into a novel graph with weight 0.5. When all weights of edges are analyzed, the Laplacian of the spectrum of the graph is analyzed [7, 14] in order to evaluate the presence of nearly disconnected components. If the graph presents nearly disconnected components, the algorithm stops, alternatively a more stringent threshold  $k$  is generated as well as a novel graph.

### 3.1 Pruning Semantic Similarity Network

In detail, the pruning algorithm examines each node  $i$  of input graph  $G_{ssu}$  and stores all weights of the adjacent edges. After this step it determines a local threshold. Then, the algorithm inserts the node  $i$  and all the adjacent ones in to final graph output of pruning  $G_{pr}$ . The algorithm analyzes each edge adjacent to  $i$  and inserts into  $G_{pr}$  those with weight greater than the determined local threshold. If the considered edge is not present in  $G_{pr}$ , the edge will have weight 0.5, otherwise the weight of the edge is set to 1. Finally all the nodes with degree = 1 are deleted from  $G_{pr}$ . In other words, if the edges are relevant considering the neighborhood of both nodes they will compare in the pruned graph with unitary weight while if the edges are relevant considering one node, they will compare with 0.5 weight. For instance by setting  $\alpha = 0$  the threshold  $k = \mu + \alpha \times sd$  results equal to the average of the weights of edges adjacent to node  $i \in G_{ssu}$ ,  $AVG(node_i)$ . The algorithm initially explores each node and discarded from the analysis the ones with degree 1. Then it explores the current node and their neighbors. The nodes are added in  $G_{pr}$ , and the edges in the pruned graph have a weight 0.5 or 1 according to the average of the weights of the neighbours nodes. In the last step all the nodes with zero degree are eliminated from  $G_{pr}$ , producing the final graph. The generation of pruned graph is repeated until the graph has nearly disconnected components, by analyzing the spectrum of the associated laplacian for value of threshold.

### 3.2 Thresholding Semantic Similarity Networks

We developed a novel hybrid thresholding method exploiting both local approach, that computes a different threshold for each node or group of nodes and global one, that prunes all edges with weights lower than the threshold. Moreover this hybrid thresholding method refers to spectral graph theory. The algorithm examines each node of a input graph and stores the weights of adjacent edge. Then a threshold  $k = \mu + \alpha \times sd$  is determined, where  $\alpha$  is a global variable, and  $\mu$  and  $sd$  are the average and standard deviation of all weights, i.e. local components.

Then, the algorithm compares each edge weight with threshold  $k$ : if the weight is greater than  $k$  considering the adjacent of both its nodes, it will be inserted into a novel graph with weight 1, whereas if the weight is greater than  $k$  considering only one of its adjacent nodes, it will be inserted into a novel graph with weight 0,5. When all weights of edges are analyzed, the Laplacian of the spectrum of the graph is analyzed [7, 14] in order to evaluate the presence of nearly disconnected components. If the graph presents nearly disconnected components, the algorithm stops, alternatively a more stringent threshold  $k$  is generated as well as a novel graph.

### 3.3 Pruning Semantic Similarity Network

In detail, the pruning algorithm examines each node  $i$  of input graph  $G_{ssu}$  and stores all weights of the adjacent edges. After this step it determines a local threshold. Then, the algorithm inserts the node  $i$  and all the adjacent ones in to final graph output of pruning  $G_{pr}$ . The algorithm analyzes each edge adjacent to  $i$  and inserts into  $G_{pr}$  those with weight greater than the determined local threshold. If the considered edge is not present in  $G_{pr}$ , the edge will have weight 0,5, otherwise the weight of the edge is set to 1. Finally all the nodes with degree = 1 are deleted from  $G_{pr}$ . In other words, if the edges are relevant considering the neighborhood of both nodes they will compare in the pruned graph with unitary weight while if the edges are relevant considering one node, they will compare with 0.5 weight. For instance by setting  $\alpha = 0$  the threshold  $k = \mu + \alpha \times sd$  results equal to the average of the weights of edges adjacent to node  $i \in G_{ssu}$ ,  $AVG(node_i)$ . The algorithm initially explores each node and discarded from the analysis the ones with degree 1. Then it explores the current node and their neighbors. The nodes are added in  $G_{pr}$ , and the edges in the pruned graph have a weight 0,5 or 1 according to the average of the weights of the neighbours nodes. In the last step all the nodes with zero degree are eliminated from  $G_{pr}$ , producing the final graph. The generation of pruned graph is repeated until the graph has nearly disconnected components, by analyzing the spectrum of the associated laplacian for value of threshold.

## 4 Architecture and Implementation of SSN-Analyzer

SSN-Analyzer, a Web Based Tool for Managing Semantic Similarity Networks for building and preprocessing Semantic Similarity Networks

SSN-Analyzer has been implemented using the Shiny framework and the R statistical language. The tool enables the calculation of semantic similarity from input genes/proteins dataset as well as the construction of SSN and preprocessing step. The tool provides a simple Graphical User Interface allowing the user an easy access to the tool functionalities.

Figure 1 conveys the interface where the user can execute a semantic similarity analysis. The tool requires as input data a list of proteins and the related annotations for each ones. It is possible to select the organism of genes/proteins

dataset, the ontology MF, BP, CC and a semantic similarity measure. Otherwise, once the organism is selected, there is the option to run the analysis by using all available semantic similarity measures on each ontology.

About the SSMs, the tool provides the measures implemented in R package csbl.go such as Resnik, ResnikGraSM, Lin, Lin-with the GraSM option (Lin-GraSM here after), JiangConrath, JiangConrath with the GraSM option (Jiang-ConrathGraSM here after), Relevance, Kappa, Cosine, WeightedJaccard, and Czekanowski Dice.

The output file (1)(c) is a semantic similarity matrix use to built the SSN. Figure 2 shows the interface where the user inserts a semantic similarity network in order to perform preprocessing process. Figure 3 conveys the output matrix and the output graph: the thresholded adjacency matrix is the result of the preprocessing analysis and the related pruned graph. Moreover, there is the option for the visualization of raw graph in order to assess the results for future analysis.

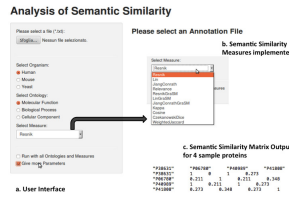


Figure 1: Semantic Similarity Analysis Interface

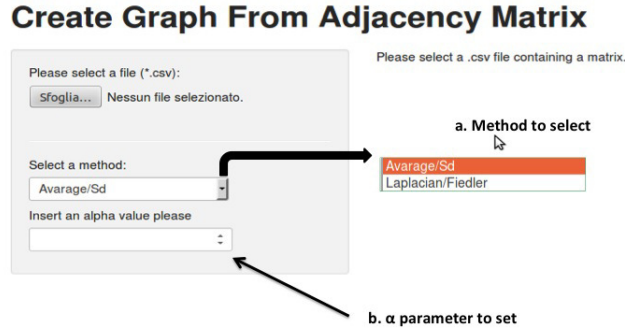


Figure 2: Graph Generation Interface

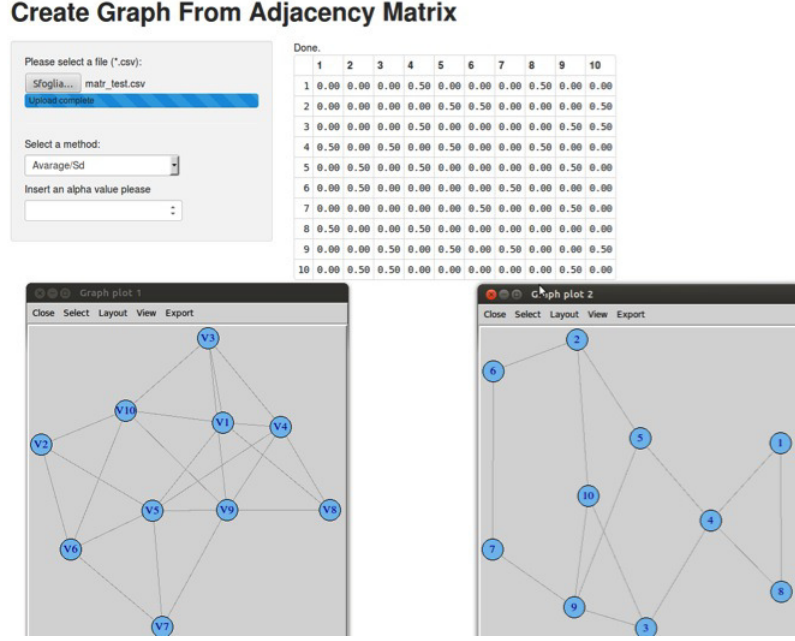


Figure 3: Thresholded Matrix. The first graph is related to no-thresholded initial matrix, the second one is resulted graph after pruning algorithm

## 4.1 Architecture

The Shiny is a R library able to extend the R functionality allowing the conversion of a R code into an interactive website user friendly. In shiny, the user can create a Graphic User Interface and publish it on web page through the extension Shiny-server. The Shiny web interfaces are automatically loaded in live mode, thus, the users can be insert any additional input parameter and it is not necessary to reload the browser page. Furthermore, several libraries are integrate very intuitive widgets optimized for Shiny, in this way the user can make simple and functional graphical interface. An R Shiny project consists of two scripts, ui.R (user-interface) and server.R (server-side).

Server.R script contains the R code, while ui.R Script defines the GUI that is displayed in the browser. In ui.R headerPanel (), sidebarPanel (), Mainpanel () are definable, and the user can build the inputs and the output. When the server.R is created it is necessary to define the logic of the server and make functional the various input and output stated in ui.R. In server.R the user imports the necessary libraries, creates the functions and imports the inputs. Then, the user assigns the outputs in order to link the results of the file server and view them in the GUI. Several libraries ad hoc can be integrated allowing the specific analysis according to own data. An example of Shiny project for a Web Based Tool for Thresholding Semantic Similarity Networks, which we

developed, is reported below:

```
library(shiny)

shinyUI(pageWithSidebar(
  headerPanel("Analysis of Semantic Similarity"),
  sidebarPanel(
    fileInput("file1", "Please select a file (*.txt)"),
    accept=c("text/text", "text/comma-separated-values,text/plain", ".txt"),
    tags$hr(),
    radioButtons("cavia", "Select Organism:",
      c("Human", "Mouse", "Yeast"),
      "Human"),
    radioButtons("ontology", "Select Ontology:",
      c("Molecular Function", "Biological Process", "Cellular Component"),
      "Molecular Function"),
    selectInput("measures", "Select Measure:",
      list("Resnik", "Lin", "JiangConrath", "Relevance", "ResnikGraSM", "LindGraSM", "JiangConrathGraSM", "Kappa", "Cosine", "CzekanowskiDice", "WeightedJaccard"), "Resnik"),
    tags$hr(),
    checkboxInput("all", "Run with all Ontologies and Measures", FALSE),
    checkboxInput("more", "Give more Parameters", FALSE)
  ),
  mainPanel(
    h3(textOutput("out"))
  )
)
```

Figure 4: ui.R Script Example

```
library(shiny)
library(caml.go)

shinyServer(function(input, output) {
  misuraSS <- function() {
    {
      fileIn <- input$file1
      ont <- switch(input$ontology,
        "Molecular Function" = "MF",
        "Biological Process" = "BP",
        "Cellular Component" = "CC"
      )

      mea <- switch(input$measures,
        "Resnik" = "Resnik",
        "Lin" = "Lin",
        "JiangConrath" = "JiangConrath",
        "Relevance" = "Relevance",
        "ResnikGraSM" = "ResnikGraSM",
        "LindGraSM" = "LindGraSM",
        "JiangConrathGraSM" = "JiangConrathGraSM",
        "Kappa" = "Kappa",
        "Cosine" = "Cosine",
        "CzekanowskiDice" = "CzekanowskiDice",
        "WeightedJaccard" = "WeightedJaccard"
      )

      tax <- switch(input$cavia,
        "Human" = TAXONOMY.HUMAN,
        "Mouse" = TAXONOMY.MOUSE,
        "Yeast" = TAXONOMY.YEAST
      )

      parte2 <- switch(input$ontology,
        "Molecular Function" = "-MF",
        "Biological Process" = "-BP",
        "Cellular Component" = "-CC"
      )
    }

    parte3 <- switch(input$measures,
      "Resnik" = "-resnik",
      "Lin" = "-lin",
      "JiangConrath" = "-jc",
      "Relevance" = "-rel",
      "ResnikGraSM" = "-resnikgrsm",
      "LindGraSM" = "-lingrsm",
      "JiangConrathGraSM" = "-jogrsm",
      "Kappa" = "-kappa",
      "Cosine" = "-cosine",
      "CzekanowskiDice" = "-cd",
      "WeightedJaccard" = "-wj"
    )

    set.prob.table(organism=tax, type="similarity")
    NomeFile <- basename(fileIn$name)

    sf.noext = sub("[.](.*)$", "", NomeFile, perl=TRUE)
    ent <- entities.from.text(fileIn$datapath)
    ris <- entity.sim.many(ent, ont, mea)
    ris=sound(ris, digits = 3)
    destinationfile <- paste("/home/phguzzi/scrivania/SS_TESTS/",
      sf.noext, parte2, parte3, ".txt", sep="")
    write.table(ris, destinationfile, sep = " ")
  }

  output$out <- renderText({
    fileIn <- input$file1
    if(is.null(fileIn))
      return("Please select an Annotation File")
    else{
      misuraSS()
      return("Done.")
    }
  })
})
```

Figure 5: server.R Script Example

## 5 Conclusions

Gene ontology and annotations are widely used in bioinformatics especially in the quantitative comparison of genes and proteins functions leveraging on the measures of semantic similarity between two or more functional genes/proteins, defined by the GO terms associated with them. Semantic Similarity Networks of proteins are a powerful instrument for biological research, especially in analysis of organisms on a system scale. Due to semantic similarity values, that



weigh the relations among proteins, the SSNs may contain meaningless edges; a preprocessing step may be necessary in order to improve the SSNs performance. For example, thresholding algorithms prune the weighted edge of graph according to a set threshold. thus they may be pruned using thresholding algorithms. In this paper we presented SSN-Analyzer, a Web Based Tool for Managing Semantic Similarity Networks able to perform a semantic similarity analysis on proteins/genes of specific organisms and build SSNs. Furthermore the tool is able to apply a novel thresholding technique, that we developed, on SSNs in order to achieve more meaningful information. The tool can be help the users to perform easily semantic similarity testing and improve the results for future analysis on SSNs.

## Acknowledgments

This work has been partially founded by project PON Smartcities DICET-INMOTO-ORCHESTRA PON04a2.D.

## References

- [1] P. Bertolazzi, M. E. Bock, and C. Guerra. On the functional and structural characterization of hubs in protein–protein interaction networks. *Biotechnology advances*, 31(2):274–286, 2013.
- [2] O. Bodenreider, M. Aubry, and A. Burgun. Non-lexical approaches to identifying associative relations in the gene ontology. *Pac Symp Biocomput*, pages 91–102, 2005.
- [3] M. Cannataro, P. H. Guzzi, and A. Sarica. Data mining and life sciences applications on the grid. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(3):216–238, 2013.
- [4] Y.-R. Cho, W. Hwang, M. Ramanathan, and A. Zhang. Semantic integration to identify overlapping functional modules in protein interaction networks. *BMC bioinformatics*, 8(1):265, 2007.
- [5] G. O. Consortium. The gene ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32(Database issue):258D–261, Jan. 2004.
- [6] F. M. Couto, M. J. Silva, and P. M. Coutinho. Measuring semantic similarity between gene ontology terms. *Data & Knowledge Engineering*, 61(1):137–152, Apr. 2007.
- [7] C. H. Q. Ding, X. He, and H. Zha. A spectral method to separate disconnected and nearly-disconnected web graph components. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 275–280, New York, NY, USA, 2001. ACM Press.

- [8] P. Guzzi, M. Mina, C. Guerra, and M. Cannataro. Semantic similarity analysis of protein data: assessment with biological features and issues. *Briefings in bioinformatics*, 13(5):569–585, 2012.
- [9] P. H. Guzzi, M. Mina, C. Guerra, and M. Cannataro. Semantic similarity analysis of protein data: assessment with biological features and issues. *Briefings in Bioinformatics*, 13(5):569–585, Sept. 2012.
- [10] D. Huang, B. Sherman, Q. Tan, J. Collins, W. G. Alvord, J. Roayaei, R. Stephens, M. Baseler, H. C. Lane, and R. Lempicki. The DAVID gene functional classification tool: a novel biological module-centric algorithm to functionally analyze large gene lists. *Genome Biology*, 8(9):R183+, Sept. 2007.
- [11] D. Lin. An information-theoretic definition of similarity. *Morgan Kaufmann, San Francisco, CA*, pages 296–304, 1998.
- [12] X. Ma and L. Gao. Biological network analysis: insights into structure and functions. *Briefings in functional genomics*, 11(6):434–442, 2012.
- [13] D. Martin, C. Brun, E. Remy, P. Mouren, D. Thieffry, and B. Jacq. GO-ToolBox: functional analysis of gene datasets based on gene ontology. *Genome biology*, 5(12):R101+, 2004.
- [14] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001.
- [15] G. P and M. M. Investigating bias in semantic similarity measures for analysis of protein interactions. In *Proceedings of 1st International Workshop on Pattern Recognition in Proteomics, Structural Biology and Bioinformatics (PR PS BB 2011)*, pages 71–80, 13th September 2011 2012.
- [16] C. Pesquita, D. Faria, A. O. Falcão, P. Lord, and F. M. Couto. Semantic similarity in biomedical ontologies. *PLoS computational biology*, 5(7):e1000443, July 2009.
- [17] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy, Nov. 1995.
- [18] A. Schlicker, F. Domingues, J. Rahnenfuhrer, and T. Lengauer. A new measure for functional similarity of gene products based on gene ontology. *BMC Bioinformatics*, 7(1):302+, June 2006.
- [19] G. Su, A. Kuchinsky, J. H. Morris, F. Meng, et al. Glay: community structure analysis of biological networks. *Bioinformatics*, 26(24):3135–3137, 2010.

- [20] X. Zhu, M. Gerstein, and M. Snyder. Getting connected: analysis and principles of biological networks. *Genes & development*, 21(9):1010–1024, 2007.